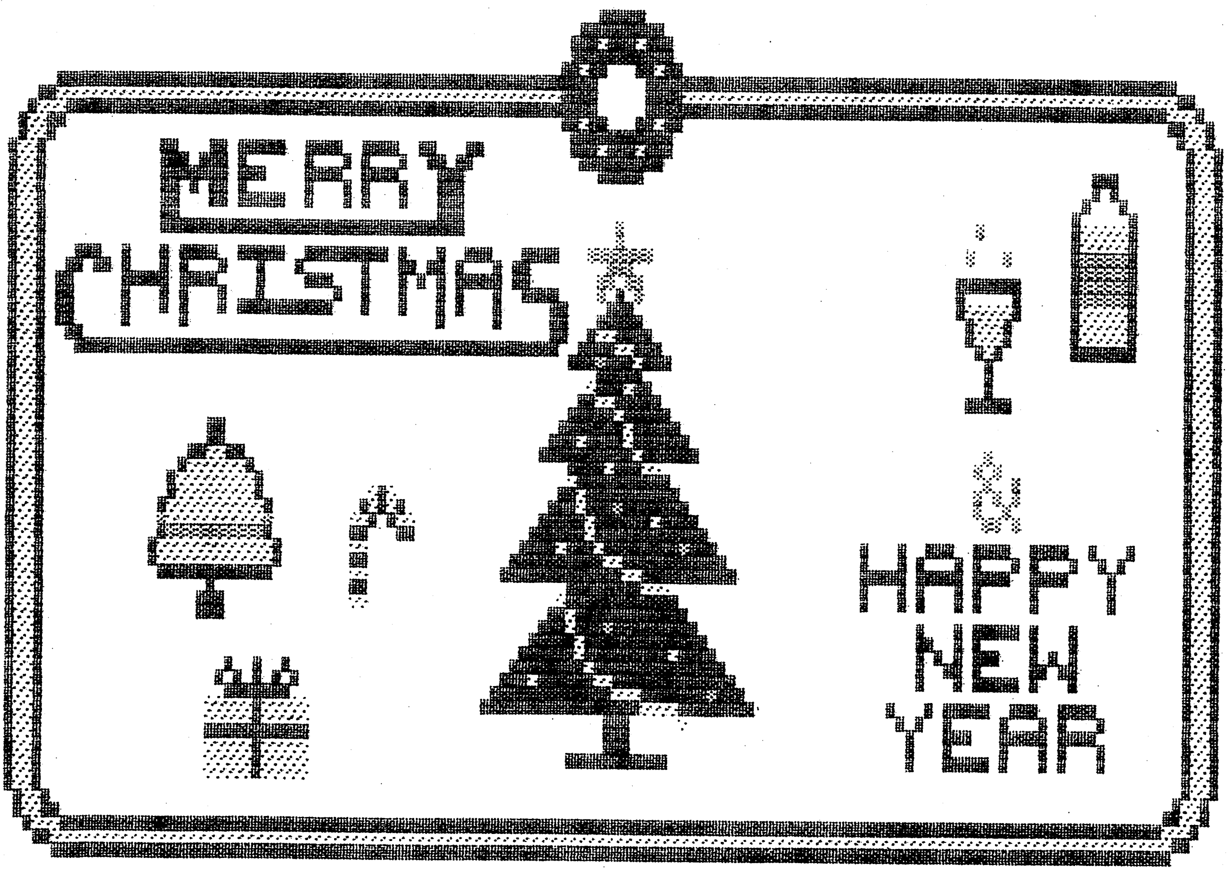
DECEMBER 1992 ISSUE # 41

HUNTER VALLEY VZ 200 300 JOURNAL



PRODUCED BI-MONTHLY BY H.V.VZ.U.G.
A NON PROFIT ORGANIZATION

FRONT COVER

JASON OAKLEY HAS BEEN BUSY ON THE FRONT COVER ONCE MORE WHICH RELATES TO THE COMING FESTIVE SEASON INSIDE. THANKS JASON.

—** MERRY CHRISTMAS AND HAPPY NEW TEAR **—

GET WELL SOON HARRY HUGGINS

ON BEHALF OF ALL VZ USERS AND MYSELF WE WISH HARRY HUGGINS A SPEEDY AND FULL RECOVERY FROM A RECENT STROKE. I BELIEVE HARRY IS COMING TO NSW (PREMIER STATE) TO RECUPERATE. SAFE JOURNEY HARRY.

CONGRATULATIONS MR & MRS HICKMAN

PETER AND DONNA ARE THE PROUD PARENTS OF A NEW BABY BOY (THAT IS TWO BOYS AND TWO GIRLS NOW). WE WISH THEM ALL GOOD HEALTH, WEALTH AND HAPPINESS FOR THE FUTURE.

GOOD NEWS FOR A CHANGE

I'VE FINALLY AQUIRED THE BLOCK OF LAND I WAS AFTER. PLANS HAVE BEEN SUBMITTED TO COUNCIL WITH BUILDING TO START SOMETIME IN FEBRUARY. MY NEW ADDRESS WILL BE: <u>35 TIGHES TCE</u> <u>TIGHES HILL</u> <u>2297</u>

MY ELDEST DAUGHTER AND SON IN LAW ARE EXPECTING MY SECOND GRANDCHILD LATE JANUARY. THE BEST NEWS IS THAT THEY ARE MOVING TO NECASTLE SOMETIME IN MARCH. I WILL BE ABLE TO SEE MY GRANDCHILDREN MORE OFTEN. IT SURE BEATS TRAVELLING TO MELBOURNE (BRRR).

PUBLIC DOMAIN LISTING

RUSSELL HARRISON
- XB - EXTENDED BASIC/DOS.
- XB2 - EXTENDED BASIC/DOS & SOURCE CODE.
- FASTDISK - IMPROVED DISK FORMATTER & SOURCE CODE.
- WORDPRO - DISK VERSION OF E & F TAPE WORD PROCESSOR.
- 8K DOS EPROM - IMPROVED VERSION OF STANDARD DOS ROM.

LESLIE MILBURN
- QW3.3 & QW4.2.2 - BASIC WORD PROCESSORS
- QWII.4.7 - 64K WORD PROCESSOR
- DISKOPS 1-4 - EDITOR ASSEMBLER.
- DISKOPS 6 - 64K EDITOR ASSEMBLER (EXTRA FUNCTIONS).

JASON OAKLEY
- DISKMAG 1 TO 4 - VZ PUBLICATION ON DISK.

ROBERT QUINN
- DDATA & TDATA - DISK & TAPE DATABASES (LOTS OF EXTRA FUNCTIONS).

PETER HICKMAN
- VZ M/C DISASSEMBLER - OBJECT AND SOURCE CODE.
- VZ MODEM SOFTWARE - OBJECT AND SOURCE CODE.
- SEE PAGE 19 FOR DESCRIPTION OF SOFTWARE AND ADDRESS.

STEVE FAITH
- VZ DISK MENU - OBJECT AND SOURCE CODE.
- SINGLE TRACK COPIER - OBJECT AND SOURCE CODE.

NOTE: CONTACT THE EDITOR FOR MORE INFORMATION ON ABOVE PROGRAMS, CONTACT ADDRESSES, ETC. ALTHOUGH ALL ABOVE ARE PUBLIC DOMAIN, A NOMINAL CHARGE WILL APPLY TO COVER COSTS LIKE DISKS, POST AND PACKING.

AUTHORS: IF YOU WOULD LIKE TO DECLARE YOUR <u>WORK PUBLIC DOMAIN</u> AS WELL THEN INFORM THE EDITOR PLEASE.

```
        CP    40
        JR    Z,EXAX
        LD    (IY+18),A
        XOR   A
        JR    EXA2
EXAX    POP   HL
        OR    A
        RET
;****************************
;DSCT: THIS FUNCTION
;      DISPLAYS THE CURRENT
;      SECTOR IN THE INPUT
;      BUFFER ON THE SCREEN.
DSCT    LD    H,(IY+50)
        LD    L,(IY+49)
        LD    DE,28672
        LD    BC,128
        LDIR
        LD    HL,28832
        LD    (7820H),HL
        LD    HL,DMSG
        CALL  PMSG
        LD    C,(IY+18)
        CALL  DHEX
        LD    A,COMA
        CALL  PCHR
        LD    C,(IY+17)
        CALL  DHEX
        LD    A,13
        CALL  PCHR
        RET
DMSG    EQU   $
:1
*CURRENT TRACK,SECTOR = *
        DEFB 0
;****************************
;DHEX: THIS DISPLAYS A
;      BINARY NO STORED IN
;      THE C REG AS AN ASCII
;      HEX PAIR AT THE
;      CURRENT CURSOR POS.
DHEX    LD    A,0F0H
        AND   C
        RRCA
        RRCA
        RRCA
        RRCA
        CALL  DHX1
        LD    A,0FH
        AND   C
        CALL  DHX1
        RET
DHX1    ADD   A,48
        CP    58
        JP    M,DHX2
        ADD   A,7
DHX2    CALL  PCHR
        RET
;****************************
;DIS: THIS FUNCTION DISPLAYS
;     THE ENTIRE DIRECTORY
;     INCLUDING START AND
;     END ADDRESSES. THE
;     FIRST CHARACTER AFTER
;     THE COMMAND INDICATES
;     WHICH FILE TYPES TO
;     DISPLAY.
DIS     CALL  GCHR
        CP    "I"
        SCF
        RET   NZ
        CALL  GCHR
        CP    "S"
        SCF
        RET   NZ
        CALL  GCHR
        JR    Z,DIS2
DIS1    PUSH  AF
        CALL  GCHR
        JP    NZ,SYER
        JR    DIS3
DIS2    XOR   A
        PUSH  AF
DIS3    EX    (SP),HL
        PUSH  HL
        LD    (IY+17),0
        LD    (IY+18),0
DIS4    DI
        CALL  PWON
        CALL  SECT
        CALL  PWOF
        EI
        LD    H,(IY+50)
        LD    L,(IY+49)
        POP   AF
        LD    DE,16
        LD    C,A
        LD    B,8
DIS5    PUSH  BC
        XOR   A
        CP    C
        JR    Z,DIS6
        LD    A,(HL)
        CP    C
        JR    Z,DIS7
        CP    0
        JR    Z,DISY
SKIP    ADD   HL,DE
        JR    DISB
DIS6    LD    A,(HL)
        CP    0
        JR    Z,DISY
        CP    1
        JR    Z,SKIP
DIS7    LD    B,10
DIS8    LD    A,(HL)
        CALL  PCHR
        INC   HL
        DJNZ  DIS8
        LD    A,32
        CALL  PCHR
        CALL  PCHR
        CALL  PCHR
        LD    B,2
```

```
DIS9 LD   C,(HL)
     CALL DHEX
     LD   A,32
     CALL PCHR
     INC  HL
     DJNZ DIS9
     LD   B,2
DISA INC  HL
     LD   C,(HL)
     CALL DHEX
     DEC  HL
     LD   C,(HL)
     CALL DHEX
     INC  HL
     INC  HL
     LD   A,32
     CALL PCHR
     DJNZ DISA
     LD   A,13
     CALL PCHR
DISB POP  BC
     DJNZ DIS5
     LD   A,(IY+17)
     INC  A
     CP   16
     JR   Z,DISX
     LD   (IY+17),A
DISC CALL WKEY
     CP   1
     JR   Z,DISX
     CP   32
     JR   NZ,DISC
     LD   A,C
     PUSH AF
     JP   DIS4
DISY POP  BC
DISX POP  HL
     OR   A
     RET
;****************************
;CHA: THIS FUNCTION CHANGES
;     THE FILETYPE OF ANY
;     FILE.
CHA  CALL GCHR
     CP   "H"
     SCF
     RET  NZ
     CALL GCHR
     CP   "A"
     SCF
     RET  NZ
     INC  HL
;SAVE RETURN ADDR ON STACK.
     LD   DE,CHA1
     PUSH DE
;SAVE BUFFER POINTER.
     PUSH HL
     LD   HL,(CSI+1)
     INC  HL
     LD   A,(HL)
     INC  HL
     LD   H,(HL)
     LD   L,A
;HL POINTS TO FUNCTION CALL
;MADE AT BEGINNING OF CSI
;FUNCTION.
     EX   (SP),HL
     RET
CHA1 OR   A
     SCF
     RET  NZ
     LD   A,(HL)
     CP   COMA
     SCF
     RET  NZ
     CALL GCHR
     JP   Z,SYER
     PUSH AF
     CALL GCHR
     JP   NZ,SYER
     EX   (SP),HL
;HL = NEW FILETYPE
     PUSH HL
     DI
     CALL PWON
     CALL SRCH
     CP   2
     LD   A,13
     JP   NZ,DERR
     IN   A,(19)
     PUSH AF
     CALL PWOF
     EI
     POP  AF
     BIT  7,A
     JR   Z,CHA2
     LD   A,4
     JP   DERR
CHA2 LD   HL,10
     EX   DE,HL
     OR   A
     SBC  HL,DE
     POP  AF
     LD   (HL),A
     DI
     CALL PWON
     CALL WRTE
     CALL PWOF
     EI
     POP  HL
     OR   A
     RET
;****************************
;REL: THIS FUNCTION
;     RELOCATES A FILE. I.E.
;     THE START ADDRESS OF A
;     FILE MAY BE CHANGED.
REL  CALL GCHR
     CP   "E"
     SCF
     RET  NZ
     CALL GCHR
     CP   "L"
     SCF
```

```
00621        RET   NZ
00622        INC   HL
00623 ;SAVE RETURN ADDR ON STACK
00624        LD    DE,REL1
00625        PUSH  DE
00626 ;SAVE BUFFER POINTER.
00627        PUSH  HL
00628        LD    HL,(CSI+1)
00629        INC   HL
00630        LD    A,(HL)
00631        INC   HL
00632        LD    H,(HL)
00633        LD    L,A
00634        EX    (SP),HL
00635        RET
00636 REL1   OR    A
00637        SCF
00638        RET   NZ
00639        LD    A,(HL)
00640        CP    COMA
00641        SCF
00642        RET   NZ
00643        INC   HL
00644        CALL  HEX
00645        RET   C
00646        PUSH  DE
00647        DEC   HL
00648        CALL  GCHR
00649        JP    NZ,SYER
00650        EX    (SP),HL
00651        PUSH  HL
00652        DI
00653        CALL  PWON
00654        CALL  SRCH
00655        CP    2
00656        LD    A,13
00657        JP    NZ,DERR
00658        IN    A,(19)
00659        PUSH  AF
00660        CALL  PWOF
00661        EI
00662        POP   AF
00663        BIT   7,A
00664        JR    Z,REL2
00665        LD    A,4
00666        JP    DERR
00667 REL2   EX    DE,HL
00668        INC   HL
00669        INC   HL
00670        POP   DE
00671        PUSH  HL
00672        PUSH  DE
00673        LD    A,(HL)
00674        INC   HL
00675        LD    H,(HL)
00676        LD    L,A
00677        RST   18H
00678        JR    Z,REL5
00679        JR    C,REL3
00680        OR    A
00681        SBC   HL,DE
00682        POP   DE
00683        EX    (SP),HL
00684        LD    (HL),E
00685        INC   HL
00686        LD    (HL),D
00687        INC   HL
00688        LD    E,(HL)
00689        INC   HL
00690        LD    D,(HL)
00691        EX    (SP),HL
00692        EX    DE,HL
00693        OR    A
00694        SBC   HL,DE
00695        JR    REL4
00696 REL3   EX    DE,HL
00697        OR    A
00698        SBC   HL,DE
00699        POP   DE
00700        EX    (SP),HL
00701        LD    (HL),E
00702        INC   HL
00703        LD    (HL),D
00704        INC   HL
00705        LD    E,(HL)
00706        INC   HL
00707        LD    D,(HL)
00708        EX    (SP),HL
00709        ADD   HL,DE
00710 REL4   EX    DE,HL
00711        POP   HL
00712        LD    (HL),D
00713        DEC   HL
00714        LD    (HL),E
00715        DI
00716        CALL  PWON
00717        CALL  WRTE
00718        CALL  PWOF
00719        EI
00720        JR    RELX
00721 REL5   POP   HL
00722        POP   HL
00723 RELX   POP   HL
00724        OR    A
00725        RET
```

AFTER ENTERING ALL THE SOURCE CODE INSERT THE FOLLOWING LINE WHICH WILL CLEAR THE SCREEN WHEN EXT12.1 IS RUN.

```
I50
CALL CLS
```

SAVE SOURCE AND OBJECT CODES. ORIGIN CAN BE ANY MEMORY LOCATION OF YOUR CHOICE.

VZ BUS MOUSE PROJECT PART 2
BY LESLIE MILBURN

THE 53 MOUSE DRIVER FUNCTIONS:

AS MENTIONED IN PART 1, THE STANDARD MICROSOFT MOUSE DRIVER PROVIDES 53 FUNCTIONS. NOT ALL OF THESE ARE CURRENTLY PROVIDED IN THE VZ BUS MOUSE DRIVER. BEFORE DETAILING EACH VZ MOUSE FUNCTION, HERE ARE THE DEFINITIONS OF SOME TERMS:-

(1) THE BUTTON REGISTER:

THIS REGISTER INDICATES WHETHER A PARTICULAR BUTTON IS PRESSED OR RELEASED.

BIT 0 CORRESPONDS TO THE LEFT BUTTON
BIT 1 CORRESPONDS TO THE RIGHT BUTTON
BIT 2 CORRESPONDS TO THE MIDDLE BUTTON (IF AVAILABLE).

IF A BIT IS SET THE THE BUTTON IS "HELD" OTHERWISE IT IS "RELEASED".

(2) BUTTON NUMBERS:

SOME FUNCTIONS REQUIRE A BUTTON NUMBER. THESE ARE AS FOLLOWS:-

0 = LEFT BUTTON
1 = RIGHT BUTTON
2 = MIDDLE BUTTON.

(3) MOUSE MOTION COUNTERS:

MOUSE MOVEMENT IS SOMETIMES REFERED TO AS MICKEYS, AND THE VALUE STORED IN A COUNTER AS THE MICKEY COUNT.

(4) PASSED AND RETURN VALUES:

ONLY THE BASIC VARIABLES WHICH ARE OF IMPORTANCE TO A PARTICULAR FUNCTION ARE LISTED. UPON RETURN, THOSE VARIABLES NOT MENTIONED MAY OR MAY NOT HAVE BEEN MODIFIED.

THE VZ MOUSE DRIVER FUNCTIONS:-

FUNCTION 0 - DRIVER RESET.

PARAMETERS: M1%=0
RETURN VALUES: NONE.

DESCRIPTION: THIS FUNCTION RESETS THE MOUSE DRIVER. THE FOLLOWING CONDITIONS ARE SET:-

MIN HORIZONTAL POS = 0
MAX HORIZONTAL POS = 31
MIN VERTICAL POS = 0
MAX VERTICAL POS = 15
CURRENT POS: HORIZ = 15
VERT. = 7
INTERNAL CURSOR FLAG = -1 (CURSOR HIDDEN)
TEXT CURSOR = ARROW
MOUSE POLLING RATE = 100 PER INTERRUPT

FUNCTION 1 - SHOW CURSOR.

PARAMETERS: M1% = 1
RETURN VALUES: NONE.

DESCRIPTION: THIS FUNCTION INCREMENTS THE INTERNAL CURSOR FLAG. IF IT EQUALS ZERO THE CURSOR IS DISPLAYED AT THE CURRENT POSITION.

FUNCTION 2 - HIDE CURSOR.

PARAMETERS: M1% = 2
RETURN VALUES: NONE.

DESCRIPTION: THIS FUNCTION REMOVES THE CURSOR FROM THE SCREEN AND DECREMENTS THE INTERNAL CURSOR FLAG.

FUNCTION 3 - GET BUTTON STATUS AND MOUSE POSITION.

PARAMETERS: M1% = 3
RETURN VALUES: M2% = BUTTON REGISTER
M3% = CURRENT HORIZONTAL POS.
M4% = CURRENT VERTICAL POS.

DESCRIPTION: THIS FUNCTION RETURNS THE STATUS OF THE MOUSE BUTTONS AND THE CURRENT MOUSE POSITION.

FUNCTION 4 - SET MOUSE POSITION.

PARAMETERS: M1% = 4
M3% = NEW HORIZONTAL POS.
M4% = NEW VERTICAL POS.
RETURN VALUES: NONE.

DESCRIPTION: THIS SETS THE MOUSE CURSOR TO THE POSITION SPECIFIED. THIS MUST BE WITHIN THE MIN AND MAX RANGES IN THE VERTICAL AND HORIZONTAL DIRECTIONS. IF NOT THE CLOSEST POINT IS CHOSEN.

FUNCTION 5 - GET BUTTON PRESS INFO.

PARAMETERS: M1% = 5
M2% = BUTTON NUMBER.
RETURN VALUES: M1% = BUTTON REGISTER
M2% = NUMBER OF BUTTON PRESSES
M3% = HORIZONTAL POS AT LAST PRESS.
M4% = VERTICAL POS AT LAST PRESS.

DESCRIPTION: THIS FUNCTION RETURNS THE BUTTON REGISTER, THE NUMBER OF PRESSES OF THE SPECIFIED BUTTON SINCE THIS FUNCTION WAS LAST CALLED AND IF THAT WAS NON-ZERO, THE POSITION OF THE MOUSE CURSOR AT THE LAST PRESS.

FUNCTION 6 - GET BUTTON RELEASE INFO.

PARAMETERS: M1% = 6
M2% = BUTTON NUMBER.
RETURN VALUES: M1% = BUTTON REGISTER
M2% = NUMBER OF BUTTON RELEASES
M3% = HORIZONTAL POS AT LAST RELEASE.
M4% = VERTICAL POS AT LAST RELEASE.

DESCRIPTION: THIS FUNCTION RETURNS THE BUTTON REGISTER, THE NUMBER OF RELEASES OF THE SPECIFIED BUTTON SINCE THIS FUNCTION WAS LAST CALLED AND IF THAT WAS NON-ZERO, THE POSITION OF THE MOUSE CURSOR AT THE LAST RELEASE.

FUNCTION 7 - SET MIN AND MAX HORIZONTAL CURSOR POS.

PARAMETERS: M1% = 7
M3% = MIN HORIZONTAL POS.
M4% = MAX HORIZONTAL POS.
RETURN VALUES: NONE.

DESCRIPTION: THIS FUNCTION SETS THE MINIMUM AND MAXIMUM HORIZONTAL POSITIONS. IF THE CURRENT HORIZONTAL POSITION IS OUTSIDE THE NEW BOUNDS, THE CURSOR IS REPOSITIONED.

FUNCTION 8 - SET MIN AND MAX VERTICAL CURSOR POS.

PARAMETERS: M1% = 8
M3% = MIN VERTICAL POS.
M4% = MAX VERTICAL POS.
RETURN VALUES: NONE.

DESCRIPTION: THIS FUNCTION SETS THE MINIMUM AND MAXIMUM VERTICAL POSITIONS. IF THE CURRENT VERTICAL POSITION IS OUTSIDE THE NEW BOUNDS, THE CURSOR IS REPOSITIONED.

FUNCTION 10 - SET TEXT CURSOR.

PARAMETERS: M1% = 10
M3% = SCREEN MASK.
M4% = CURSOR MASK.
RETURN VALUES: NONE.

DESCRIPTION: THIS FUNCTION ALLOWS THE TEXT MODE MOUSE CURSOR TO BE CHANGED. THE CHARACTER AT THE CURRENT POSITION IS ANDED WITH THE SCREEN MASK AND THE RSULT IS XORED WITH THE CURSOR MASK. THE RESULT IS DISPLAYED AT THE CURRENT POSITION.

FUNCTION 11 - READ MOUSE MOTION COUNTERS.

PARAMETERS: M1% = 11
RETURN VALUES: M3% = HORIZONTAL MICKEY COUNT
M4% = VERTICAL MICKEY COUNT.

DESCRIPTION: THIS FUNCTION RETURNS THE HORIZONTAL AND VERTICAL MICKEY COUNTS SINCE THIS FUNCTION WAS LAST CALLED.

FUNCTION 12 - SET SUBROUTINE CALL MASK AND ADDRESS.

PARAMETERS: M1% = 12
M3% = CALL MASK.
M4% = SUBROUTINE ADDRESS.
RETURN VALUES: NONE.

DESCRIPTION: THIS FUNCTION ALLOWS A SUBROUTINE TO BE CALLED WHENEVER ONE OR MORE OF THE CONDITIONS DEFINED BY THE CALL MASK OCCURS. EACH BIT IN THE CALL MASK CORRESPONDS TO A SPECIFIC CONDITION:

BIT	CONDITION
0	CURSOR POSITION CHANGED.
1	LEFT BUTTON PRESSED.
2	LEFT BUTTON RELEASED.
3	RIGHT BUTTON PRESSED.

BIT	CONDITION
4	RIGHT BUTTON RELEASED.
5	MIDDLE BUTTON PRESSED.
6	MIDDLE BUTTON RELEASED.
7-15	NOT USED.

FUNCTION 20 - SWAP SUBROUTINES.

PARAMETERS: M1% = 20
M3% = NEW CALL MASK
M4% = NEW SUBROUTINE ADDRESS.
RETURN VALUES: M3% = OLD CALL MASK.
M4% = OLD SUBROUTINE ADDRESS.

DESCRIPTION: THIS FUNCTION ALLOWS A SUBROUTINE ADDRESS AND CALL MASK TO BE SET AS IN FUNCTION 12. THE DIFFERENCE IS THAT THE OLD ADDRESS AND MASK ARE RETURNED

FUNCTION 21 - GET MOUSE DRIVER STATE STORAGE REQUIREMENTS.

PARAMETERS: M1% = 21
RETURN VALUES: M2% = BUFFER SIZE.

DESCRIPTION: THIS FUNCTION RETURNS THE SIZE REQUIRED TO STORE THE CURRENT STATE OF THE DRIVER.

FUNCTION 22 - SAVE MOUSE DRIVER STATE.

PARAMETERS: M1% = 22
M4% = BUFFER POINTER.
RETURN VALUES: NONE.

DESCRIPTION: THIS FUNCTION COPIES ALL MOUSE VARIABLES INTO THE PROVIDED BUFFER. IT IS ASSUMED THAT THE BUFFER IS BIG ENOUGH.

FUNCTION 23 - RESTORE MOUSE DRIVER STATE.

PARAMETERS: M1% = 23
M4% = BUFFER POINTER.
RETURN VALUES: NONE.

DESCRIPTION: THIS FUNCTION RESTORES THE MOUSE DRIVER STATE WHICH WAS PREVIOUSLY SAVED USING FUNCTION 22.

FUNCTION 24 - SET ALTERNATE SUBROUTINE CALL MASK AND ADDRESS

PARAMETERS: M1% = 24
M3% = CALL MASK.
M4% = SUBROUTINE ADDRESS.
RETURN VALUES: M1% = ERROR FLAG (-1 IF ERROR).

DESCRIPTION: THIS FUNCTION ALLOWS UP TO THREE SUBROUTINES TO BE SPECIFIED. EACH CALL MASK MUST BE UNIQUE. THE CALL MASK DEFINITION IS:-

BIT	CONDITION
0	CURSOR POSITION CHANGED.
1	LEFT BUTTON PRESSED.
2	LEFT BUTTON RELEASED.
3	RIGHT BUTTON PRESSED.
4	RIGHT BUTTON RELEASED.

BIT	CONDITION
5	MIDDLE BUTTON PRESSED.
6	MIDDLE BUTTON RELEASED.
7	NOT USED.
8	SHIFT KEY PRESSED.
9	CTRL KEY PRESSED.
10	ALT KEY PRESSED.
11-15	NOT USED.

TO BE CONTINUED (YET AGAIN)!

HI-RES GRAPHICS GEOMETRIC PLOTTING BY BOB KITCH

THE FOLLOWING PROGRAM IS A SIMPLE LINE PLOTTING ROUTINE USING THE HI-RES GRAPHICS SCREEN. IT WAS WRITTEN TO TRY AND DEMONSTRATE HOW PROGRAMMING SKILLS CAN BE IMPROVED BY FOLLOWING A FEW SIMPLE GUIDELINES. IT IS A PLEA FOR MORE READABLE BASIC PROGRAMS.

UNFORTUNATELY PUBLISHED PROGRAMS, IN MAGAZINES, ARE GENERALLY POOR EXAMPLES OF HOW TO DEVELOP GOOD PROGRAMMING STYLE. A NUMBER OF US MAY HAVE TAKEN THE TROUBLE TO ENTER A LISTING FROM A MAGAZINE - BUT UPON RUNNING THE CODE HAVE FOUND THAT ALL IS NOT WELL!

A LONG, TEDIOUS AND FRUSTRATING SESSION OF UNDERSTANDING THE POORLY CONSTRUCTED CODE FOLLOWS. OFTEN THIS REQUIRES THAT THE TWISTS AND TURNS OF THE "LOGICAL SPAGHETTI" BE UNRAVELLED BEFORE DEBUGGING CAN COMMENCE. A USUAL REMEDY IS TO RE-WRITE THE PROGRAM FROM SCRATCH.

THE PROGRAM LINPLOT IS WRITTEN USING THE FOLLOWING GUIDELINES -

1. CLEARLY CODED AND SET OUT - AN ENORMOUS HELP TO UNDERSTANDING.

2. THE PROGRAM IS STRUCTURED - A GOOD ALGORITHM IS SELECTED AND THE PROGRAM "FLOWS" THROUGH INITIALIZATION, TO INPUT, PROCEDURE AND OUTPUT SECTIONS.

3. LOOPS ARE INDENTED FOR EASE OF IDENTIFICATION AND NESTING.

4. NAMING OF VARIABLES IS MEANINGFUL TO ASSIST MAINTENANCE AND DEBUGGING.

5. INTEGER STORAGE IS USED WHERE APPROPRIATE.

6. NO ABBREVIATED FORMS OF BASIC STATEMENTS ARE USED.

7. REMARKS ARE LIBERALLY SPRINKLED THROUGHOUT TO AID CLARITY.

8. ERROR CAPTURE AND RANGE CHECKING ON ALL INPUT VARIABLES PREVENTS THE PROGRAM FROM CRASHING.

CLEAR READABLE CODE IS MORE IMPORTANT THAN THE EXECUTION SPEED OR STORAGE REQUIREMENTS OF A PROGRAM - INTERPRETED BASIC RUNS LIKE A TIRED SNAIL IN ANY CASE!

THESE GUIDELINES SHOULD LEAD TO CODE THAT IS EASIER TO READ, UNDERSTAND AND DEBUG. THIS LEADS TO EASIER MAINTENANCE, UPDATING OR EXPANSION OF YOUR ROUTINES AS YOUR PROGRAMMING SKILLS DEVELOP.

```
010 '*****************************************************************
020 '*    PLOT A SET OF UP TO 20 LINES USING THE HI-RES SCREEN    *
030 '*                  BY R.B.KITCH  22/10/85                    *
040 '*****************************************************************
090 '
100 '***DIMENSION STORAGE VECTORS X% & Y%.
110 DIM X%(20),Y%(20):CLS:'***VECTORS TO HOLD END CO-ORDS.
115 '
120 '***ACCEPT INPUT AND CHECK.
130 INPUT"HOW MANY LINES - MAX 20 ";LN%
140 IF LN%<1 OR LN%>20 THEN GOTO 130
150 FOR I%=0 TO LN%       :'***LOOP FOR LN%+1 X-Y POINTS.
160    INPUT"ENTER X-VAL 0-127 ";X%(I%)
170    IF X%(I%)<0 OR X%(I%)>127 THEN GOTO 160:'***CHECK ON SCRN
180    INPUT"ENTER Y-VAL 0- 63 ";Y%(I%)
190    IF Y%(I%)<0 OR Y%(I%)> 63 THEN GOTO 180:'***CHECK ON SCRN
200 NEXT I%
290 '
300 '***SET UP SCREEN AND MAIN PLOTTING LOOP.
310 MODE(1)               :'***SWITCH SCREEN TO HI-RES.
320 FOR I%=0 TO LN%-1     :'***ASSIGN MAIN LOOP FOR LN% LINES.
330    X1%=X%(I%):X2%=X%(I%+1):'***ASSIGN END POINTS TO TEMP VAR
340    Y1%=Y%(I%):Y2%=Y%(I%+1):'***ASSIGN END POINTS TO TEMP VAR
350 '***ARE THE POINTS THE SAME?
360    IF X1%<>X2% OR Y1%<>Y2% THEN GOTO 410
370    SET(X1%,Y1%)        :'***END POINTS ARE THE SAME SO PLOT.
380    GOTO 710
390 '
400 '***CALCULATE X AND Y DIFFERENCE.
410    DX%=X2%-X1%:DY%=Y2%-Y1%:'***CHANGE IN X & Y DIRECTIONS.
420 '***SEE WHICH IS LARGER.
430    IF ABS(DX%)>ABS(DY%) THEN GOTO 610
490 '
500 '***INCREMENT IY OR ALONG Y-AXIS.
510    YS%=SGN(DY%):DG=DX%/DY%:'***SIGN OF STEP AND GRADIENT.
520    XO=X1%+0.5          :'***X-AXIS OFFSET.
530    FOR IY%=Y1% TO Y2% STEP YS%:'***INITIALIZE LOOP.
540       TP=(IY%-Y1%)*DG+XO:'***TEMP REAL X-VALUE.
550       IX%=INT(TP)      :'***INTEGER X-VALUE.
560       SET(IX%,IY%)
570    NEXT IY%
580    GOTO 710            :'***PICK UP ANOTHER LINE.
600 '***INCREMENT IX OR ALONG X-AXIS.
610    XS%=SGN(DX%):DG=DY%/DX%:'***SIGN OF STEP AND GRADIENT.
620    YO=Y1%+0.5          :'***Y-AXIS OFFSET.
630    FOR IX%=X1% TO X2% STEP XS%:'***INITIALIZE LOOP.
640       TP=(IX%-X1%)*DG+YO:'***TEMP REAL Y-VALUE.
650       IY%=INT(TP)      :'***INTEGER Y-VALUE.
660       SET(IX%,IY%)
670    NEXT IX%
690 '
700 '***END LOOP FOR LINE.
710 NEXT I%:SOUND 28,6    :'***END LOOP.
720 AN$=""
730 AN$=INKEY$:AN$=INKEY$ :'***PAUSE FOR ANY KEYSTROKE.
740 IF AN$="" THEN GOTO 730
790 '
800 '***GO AGAIN?
810 CLS:PRINT"   (E) TO EXIT":'***SCREEN MESSAGE OR MENU.
820 PRINT"   (P) TO PLOT AGAIN"
830 PRINT"   (N) FOR NEW POINTS":PRINT
```

```
840 INPUT AN$                 :'***ACCEPT RESPONSE.
850 AN$=LEFT$(AN$,1)          :'***CLEAN IT UP.
860 IF AN$="E" THEN STOP      :'***LOGICAL END TO PROGRAM.
870 IF AN$="P" THEN GOTO 310:'***GO BACK AND PLOT AGAIN.
880 IF AN$="N" THEN GOTO 130:'***GO BACK FOR MORE INPUT.
890 GOTO 810                  :'***WRONG RESPONSE.
900 END
```

LET'S INVESTIGATE SOUND ON THE VZ PART IV BY BOB KITCH

FOR THE NEXT SESSION ON SOUND GENERATION ON THE VZ, I WILL DETAIL SOME ARTICLES ON PERIPHERAL DEVICES THAT CAN BE CONNECTED TO THE VZ. THESE CAN GREATLY EXPAND THE APPEAL OF THE MACHINE AND ENHANCE YOUR INTEREST IN THE VZ. (NOT TO MENTION THE ENTHUSIASM THAT OTHERS WILL GET FOR THE COMPUTER.)

THERE ARE TWO TYPES OF "NOISE MAKING" PERIPHERALS. THESE ARE VOICE AND SOUND SYNTHESIS I.C. CHIPS. THESE ARE ALTERNATE AND NOVEL FORMS OF OUTPUT, TO THAT OBTAINED FROM THE SCREEN OR PRINTER, WHEN ONE HAS TIRED OF THESE ENTIRELY VISUAL FORMS OF OUTPUT. MUSIC SYNTHESIS EXCEEDS THE CAPABILITIES OF THE VZ'S INBUILT PIEZO-SPEAKER.

A NUMBER OF CIRCUITS AND PROJECTS HAVE APPEARED IN THE MAGAZINES OVER THE PAST FEW YEARS. THIS ARTICLE BRIEFLY IDENTIFIES THESE FOR THOSE WHO MAY WISH TO BUILD A BOARD OR ALTERNATIVELY REGISTER INTEREST WITH ME SO THAT WE CAN MAKE AVAILABLE THESE PERIPHERALS PLUS SOME OFF-THE-SHELF SOFTWARE.

IMAGINE THE BLOCKBUSTING USE OF VOICE AND MUSIC SYNTHESIS IN GAMES OR APPLICATIONS FOR THE VZ.

A COUPLE OF INTRODUCTORY ARTICLES ON SPEECH SYNTHESIS APPEARED IN BYTE SEP. 84, P.337 AND ITEC #26, P.812. THESE PROVIDE GOOD BACKGROUND.

	MAGAZINE	DATE	NAME	CHIP	INTERFACE	SOFTWARE
VOICE	EA	OCT. 82 APR. 83	COMPU- VOICE	VOTRAX SC-01	CENTRONICS	YES
	APC	DEC. 84	DIY SYNTH.	SC-01	CENTRONICS	YES
	ETI	JAN. 85 APR. 86	CHATTER -BOX	SC-01	CENTRONICS	YES
	ETI	MAR. 86	TALKING VZ-200	GI SP0256	PARALLEL	NO
	AEM	FEB. 86	PROJECT 4505	GI SP0256	CENTRONICS	YES
	PE	MAR. 85 JUN. 85	BBC	GI SP0256	PARALLEL	YES
	PE	JAN. 86	SPECTRUM			YES
SOUND	APC	NOV. 84	DIY SYNTH.	TI SN76496	CENTRONICS	YES
	EA	AUG. 83	COMPU- MUSE	TI SN76489	CENTRONICS	YES

SO IF YOU ARE TIRED OF READING OUTPUT FROM YOUR COMPUTER, WHY NOT TRY LISTENING INSTEAD?

WHEN THE DISK DRIVE FIRST BECAME AVAILABLE FOR THE VZ 200/300 COMPUTERS THINGS WERE SIMPLE AND THERE WAS NO CONFUSION AS THERE WERE ONLY 3 FILETYPES TO WORRY ABOUT, EG:

```
T:FILENAME 7AE9 XXXX XXXX - TEXT FILE   - (BASIC PROGRAM)

B:FILENAME XXXX XXXX XXXX - BINARY FILE - (MACHINE/OBJECT CODE)
B:FILENAME 7000 7800 0800 - BINARY FILE - (HI-RES SCREEN)
B:FILENAME C000 FFFF 4000 - BINARY FILE - (MEMORY BLOCKS)

D:FILENAME 0000 0000 0000 - DATA FILE   - (PROGRAM GENERATED)
```

AS VZ USERS STARTED WRITING PROGRAMS FOR DISK DRIVE USE AND QUITE OFTEN WITHOUT CONSULTATION WITH OTHERS THEY INTRODUCED OTHER FILETYPES AND THE CONFUSION AND INCOMPATIBILITY BEGAN. BELOW IS A LIST OF THE NEW FILETYPES, THEIR USES AND THEIR AUTHORS.

DISK ED/ASS SOURCE CODE FILETYPES

```
A:FILENAME A280 XXXX XXXX - EDITOR ASSEMBLER - RH - RUSSELL HARRISON
S:FILENAME A280 XXXX XXXX - EDITOR ASSEMBLER - DM - DAVE MITCHELL
S:FILENAME A280 XXXX XXXX - EDITOR ASSEMBLER - MH - MARK HARWOOD
           A280 XXXX XXXX - VARIANT          - BG - BRIAN GREEVE
           A280 XXXX XXXX - VARIANT          - PH -PETER HICKMAN
W:FILENAME A813 XXXX XXXX - DISKOPS ED/ASS.  - LM - LESLIE MILBURN
```

THERE ARE FOUR BASIC DISK VERSIONS AND TWO VARIANTS OF DICK SMITH'S EDITOR ASSEMBLER WHOSE SOURCE CODE FILES ARE NOT COMPATIBLE WITH EACH OTHER. AS YOU'LL NOTE THERE ARE THREE DIFFERENT FILETYPE'S, A, S AND W AND TWO DIFFERENT START ADDRESSES, (A280 & A813).

DISK WORD PROCESSOR FILETYPES

```
W:FILENAME XXXX D000 XXXX - PATCH 3.3  - DM - DAVE MITCHELL
W:FILENAME XXXX XXXX XXXX - WORDPRO    - RH - RUSSELL HARRISON
F:FILENAME 0000 FFFF FFFF - QUICKWRITE - LM - LESLIE MILBURN
```

AGAIN INCOMPATIBILITY IS THE NAME OF THE GAME WITH DIFFERENT FILETYPE'S AND START AND END ADDRESSES. TO ADD MORE CONFUSION THERE ARE TWO WORD PROCESSOR AND ONE EDITOR ASSEMBLER SHARING A W:FILETYPE.

CONVERTING SOURCE CODE FILES

TO DENOTE THE VARIOUS EDITOR ASSEMBLERS AND THEIR SOURCE CODE, INITIALS WILL BE USED FOR COMPARISON PURPOSES. SEE LAST TWO CHARACTERS IN FILENAME BELOW. I'LL USE LESLIE MILBURN'S EXT12.1 (PART II IN THIS ISSUE) AS AN EXAMPLE.

```
S:EXT-DM   01 00 A280 CF0A 2C8A
S:EXT-MH   01 00 A280 CF0C 2C8C
A:EXT-RH   01 00 A280 CF0C 2C8C
W:EXT-LM   01 00 A813 D4A0 2C8D
```

I STARTED OUT WITH W:EXT-LM SOURCE CODE FILE AND AFTER CONVERSION ARRIVED AT THE REST. CONVERTING CAN BE AS SIMPLE AS CHANGING FILETYPE AND OR START/END ADDRESSES. I USED LESLIE MILBURN'S EXT12.1 DOS UTILITY AS THE MOST SUITABLE FOR THE PURPOSE AS IT HAS TWO OF THE COMMANDS REQUIRED. THEY ARE:

1) CHA"FILENAME",X
2) REL"FILENAME",XXXX

TO CHANGE FILETYPE ACTIVATE EXT12.1 AND TYPE IN:

```
CHA"EXT.LM",S    <RETURN>
THIS WILL CHANGE (W) FILETYPE TO (S)
```

TO CHANGE START AND END ADDRESSES TYPE IN:

```
REL"EXT.LM",A280    <RETURN>
```

NOTE 1: EXT12.1 WILL AUTOMATICALLY WORK OUT NEW END ADDRESS TO CORRESPOND TO NEW START ADDRESS AND UPDATE DISK DIRECTORY.

NOTE 2: DM(S), MH(S), RH(A) AND LM(W) = THE 4 SOURCE CODE FORMATS.

AND NOW TO THE DETAILS ON HOW TO CONVERT SOURCE CODE FOR USE BETWEEN EDITOR ASSEMBLERS.

DM - WILL LOAD DM(S) AND MH(S) WITHOUT MODIFICATION.
WILL LOAD RH(A) AFTER CHANGING FILETYPE TO DM(S).
WILL LOAD LM(W) AFTER CHANGING FILETYPE TO DM(S) AND START ADDRESS FROM A813 TO A280.

MH - WILL LOAD MH(S) WITHOUT MODIFICATION.
WILL LOAD RH(A) AFTER CHANGING FILETYPE TO MH(S).
WILL LOAD DM(S) AFTER USING ASM.DM AND COMPAT ROUTINE TO CONVERT DM(S) TO MH(S) FORMAT.
WILL LOAD LM(W) AFTER CHANGING TO DM(S) FORMAT AND THEN TO MH(S) FORMAT.

RH - WILL LOAD RH(A) WITHOUT MODIFICATION.
WILL LOAD MH(S) AFTER CHANGING FILETYPE TO RH(A).
WILL LOAD DM(S) AFTER CHANGING TO MH(S) FORMAT AND CHANGING FILETYPE TO RH(A).
WILL LOAD LM(W) AFTER CHANGING TO DM(S) FORMAT AND THEN TO MH(S) FORMAT AND CHANGING FILETYPE TO RH(A).

LM - WILL LOAD LM(W) WITHOUT MODIFICATION
WILL LOAD DM(S) AFTER CHANGING START ADDRESS TO A813 AND FILETYPE TO LM(W).
WILL LOAD MH(S) AFTER FIRST CHANGING TO DM(S) FORMAT AND THEN TO LM(W) FORMAT.
WILL LOAD RH(A) AFTER FIRST CHANGING TO DM(S) AND THEN CHANGING DM(S) TO LM(W) FORMAT.

THE FOLLOWING ARE THE EDITOR ASSEMBLERS USED FOR COMPILING THIS ARTICLE AND ONCE AGAIN INITIALS ARE USED TO DENOTE AUTHORS.

```
B:ASM.DM 01 0D 7AFD A2F3 27F6
S:COMPAT 01 02 A280 A7C2 0542 - CONVERT ROUTINE TO CHANGE
                                 DM(S) FORMAT TO MH(S) FORMAT.
B:ASM.MH 06 0F 7AFD A2A3 27A6
B:ASM.LM 0C 00 7AFD BF01 4404 - DISKOPS 6, 64K VERSION.
T:ASM.RH 14 0B 7AE9 7B29 0040 - 34K VERSION.
B:ASM1   14 0C FC00 FE81 0281
B:ASM2   15 02 7AFA A301 2807 - NOTE: ASM.RH CONSISTS OF 3 FILES.
```

THE EDITOR ASSEMBLER I PREFER IS DAVE MITCHELL'S VERSION AS IT IS MORE COMPATIBLE THAN THE OTHERS AND CAN CONVERT BOTH WAYS BETWEEN ASM.MH AND ASM.DM, ED.

Z80A TIMING DIAGRAMS

THE MAIN AIM OF THE TIMING DIAGRAMS IS TO SHOW YOU WHAT HAPPENS TO THE IORQ, RD AND WR LINES WHEN THE I/O (INP OR OUT) FUNCTIONS ARE USED.

WHEN A PORT IS READ BY USING THE INP FUNCTION THE IORQ AND RD LINES GO LO WHILE WRITE LINE STAYS HI. WHEN WE WRITE TO A PORT USING THE OUT FUNCTION THE IORQ AND WR LINES GO LO WHILE THE READ LINE STAYS HI.

THIS MEANS THAT BOTH THE READ AND WRITE LINES CAN NEVER BE (1) OR (Ø) AT THE SAME TIME. WE CAN PUT THIS TO GOOD USE BY MAKING A READ (I/P) ONLY PORT OR WRITE (O/P) ONLY PORT OR A READ/WRITE (I/O) PORT.

74LS138 TRUTH TABLE

THE 74LS138, 3 OF 8 DECODER IC IS THE MOST COMMONLY FOUND DECODER USED IN VZ APPLICATIONS. IT IS AN ACTIVE LO DEVICE AND FOR IT TO WORK EN1 BAR AND EN2 BAR MUST BE LO WHILE EN3 MUST BE HI.

INPUT/OUTPUT PORT DECODING

ON PAGE 17, THE LEFT DECODER DECODES IN 16 BIT BLOCKS IN THE PORT ADDRESS RANGE OF AØ TO A127. TO MAKE IT A READ/WRITE PORT CONNECT +5V TO PIN 6, EN3 (POSITIVE ENABLE). FOR A READ ONLY PORT CONNECT THE WR LINE TO PIN 6 WHILE FOR WRITE ONLY PORT CONNECT THE RD LINE TO PIN 6.

BECAUSE WE ARE USING A POSITIVE ENABLE TO CONTROL WHAT THE PORT WILL DO WE HAVE TO INVERT THE RD AND WR LINES BY USING THE OPPOSITE. REFER TO THE TIMING DIAGRAMS FOR CLARIFICATION.

THE SECOND DECODER DECODES IN THE RANGE OF A128 TO A255. BECAUSE WE ARE USING A NEGATIVE ENABLE IT IS EASIER TO UNDERSTAND. CONNECT GND TO PIN 5, EN2 BAR FOR READ/WRITE PORT. RD TO PIN 5 FOR A READ ONLY PORT OR WR FOR A WRITE ONLY PORT.

YOU'LL NOTE BOTH DECODERS ARE NEARLY IDENTICAL, EXCEPT FOR A7 WHICH IS USED TO SELECT DECODING RANGE. IN THE FIRST DECODER A7 IS CONNECTED TO PIN 5, A NEGATIVE ENABLE. THIS MEANS ANY PORT GREATER THAN 128 CANNOT BE SELECTED. IN THE SECOND DECODER A7 IS CONNECTED TO PIN 6, A POSITIVE ENABLE WHERE A7 LOCKS OUT ANY PORT LESS THAN 128.

TECHNICAL DATA SHEET # 4

PAGE 18 SHOWS A READ/WRITE PORT, A READ ONLY PORT AND A WRITE ONLY PORT IN THE PORT ADDRESS RANGES OF AØ TO A127 AND A128 TO A255.

THE 64K RAM EXPANSION DECODER IS CONFIGURED AS A READ/WRITE PORT BUT CAN BE CHANGED TO A READ ONLY PORT OR WRITE ONLY PORT BY REMOVING GND FROM PIN 5 AN CONNECTING EITHER THE RD OR WR LINE TO PIN 5.

THE 32 BIT DECODER SHOWS IT IS CONNECTED AS A READ/WRITE PORT BUT CAN BE CHANGED TO TO READ ONLY PORT OR WRITE ONLY PORT AS SHOWN IN PREVIOUS EXAMPLE.

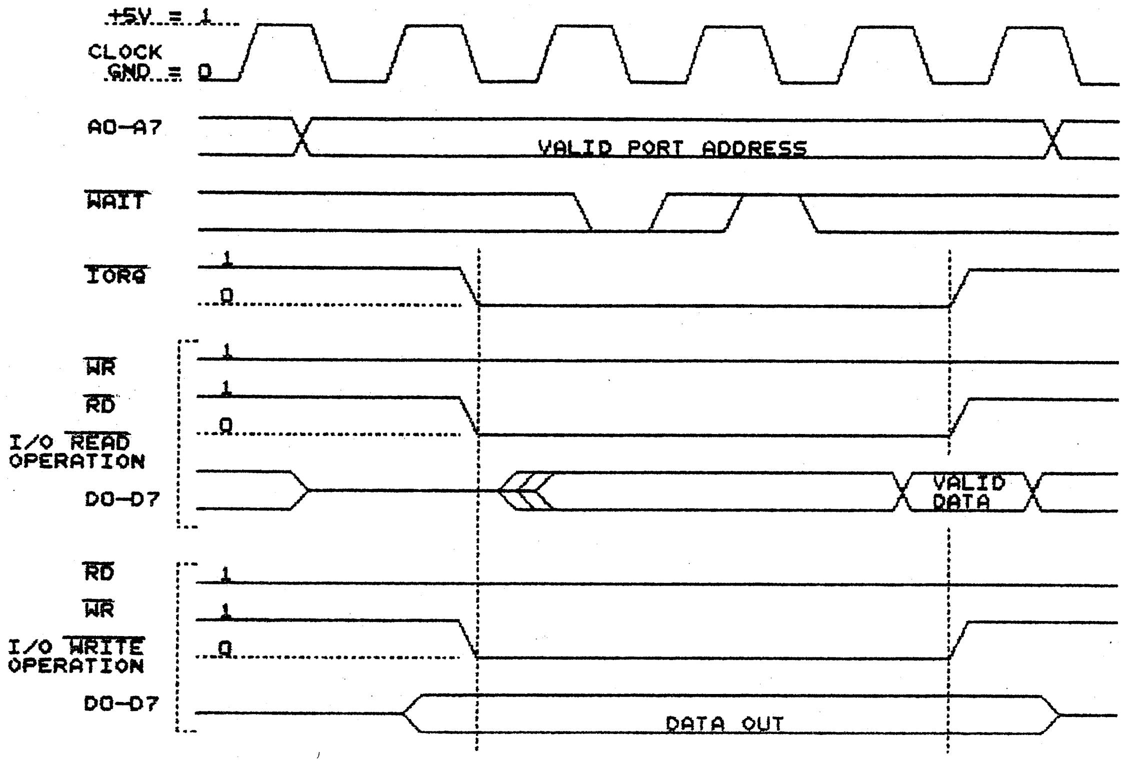
NEXT ISSUE: MEMORY DECODING.

Z80A I/O TIMING DIAGRAMS
74LS138 TRUTH TABLE
INPUT/OUTPUT PORT DECODING

TECH-3 11-12-1992

Z80A INPUT OR OUTPUT CYCLES

THE Z80A TIMING DIAGRAMS BELOW SHOWS THE TIMING FOR AN I/O READ OR FOR AN I/O WRITE OPERATION. DURING I/O OPERATIONS, THE CPU AUTOMATICALLY INSERTS A SINGLE WAIT STATE. THIS EXTRA WAIT STATE ALLOWS SUFFICIENT TIME FOR AN I/O PORT TO DECODE THE ADDRESS FROM THE PORT ADDRESS LINES.



X = DONT CARE

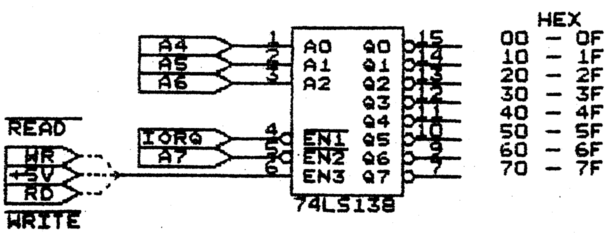
74LS138 TRUTH TABLE

1 = HI / 0 = LO

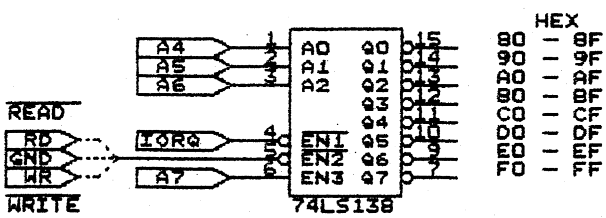
INPUTS						OUTPUTS							
ENABLE			SELECT										
$\overline{EN1}$	$\overline{EN2}$	EN3	A0	A1	A2	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7
1	X	X	X	X	X	1	1	1	1	1	1	1	1
X	1	X	X	X	X	1	1	1	1	1	1	1	1
X	X	0	X	X	X	1	1	1	1	1	1	1	1
0	0	1	0	0	0	0	1	1	1	1	1	1	1
0	0	1	1	0	0	1	0	1	1	1	1	1	1
0	0	1	0	1	0	1	1	0	1	1	1	1	1
0	0	1	1	1	0	1	1	1	0	1	1	1	1
0	0	1	0	0	1	1	1	1	1	0	1	1	1
0	0	1	1	0	1	1	1	1	1	1	0	1	1
0	0	1	0	1	1	1	1	1	1	1	1	0	1
0	0	1	1	1	1	1	1	1	1	1	1	1	0

INPUT/OUTPUT PORT DECODING BY JOE LEON

16 BIT BLOCK I/O DECODER * 00H - 7FH



16 BIT BLOCK I/O DECODER * 80H - FFH



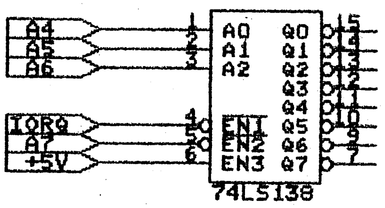
INPUT/OUTPUT PORT DECODING

TECH-4 11-12-1992

INPUT/OUTPUT PORT DECODING BY JOE LEON

16 BIT BLOCK I/O DECODER = 00H - 7FH = 000 - 127

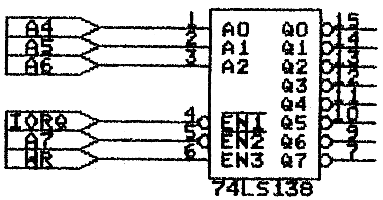
READ/WRITE



HEX	DEC
00 - 0F	00 - 15
10 - 1F	16 - 31
20 - 2F	32 - 47
30 - 3F	48 - 63
40 - 4F	64 - 79
50 - 5F	80 - 95
60 - 6F	96 - 111
70 - 7F	112 - 127

16 BIT BLOCK I/P DECODER = 00H - 7FH = 000 - 127

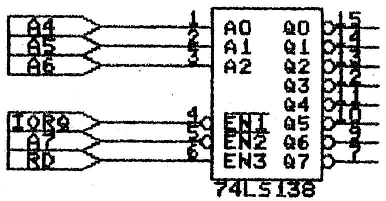
READ ONLY



HEX	DEC
00 - 0F	00 - 15
10 - 1F	16 - 31
20 - 2F	32 - 47
30 - 3F	48 - 63
40 - 4F	64 - 79
50 - 5F	80 - 95
60 - 6F	96 - 111
70 - 7F	112 - 127

16 BIT BLOCK O/P DECODER = 00H - 7FH = 000 - 127

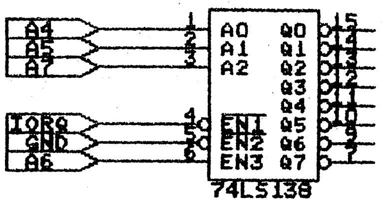
WRITE ONLY



HEX	DEC
00 - 0F	00 - 15
10 - 1F	16 - 31
20 - 2F	32 - 47
30 - 3F	48 - 63
40 - 4F	64 - 79
50 - 5F	80 - 95
60 - 6F	96 - 111
70 - 7F	112 - 127

16 BIT BLOCK I/O DECODER - 64K RAM PACK EXPANSION

READ/WRITE

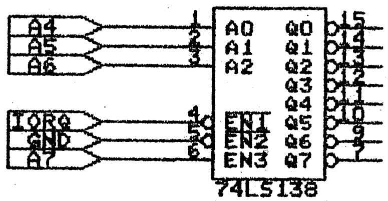


HEX	DEC
40 - 4F	64 - 79
50 - 5F	80 - 95
60 - 6F	96 - 111
60 - 6F	112 - 127
C0 - CF	192 - 207
B0 - BF	208 - 223
E0 - EF	224 - 239
F0 - FF	240 - 255

DECODED BY GARY BULLEY

16 BIT BLOCK I/O DECODER = 80H - FFH = 128 - 255

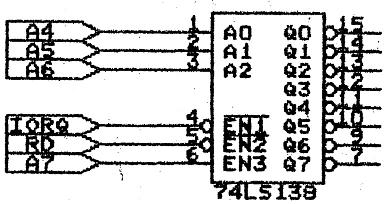
READ/WRITE



HEX	DEC
80 - 8F	128 - 143
90 - 9F	144 - 159
A0 - AF	160 - 175
B0 - BF	176 - 191
C0 - CF	192 - 207
D0 - DF	208 - 223
E0 - EF	224 - 239
F0 - FF	240 - 255

16 BIT BLOCK I/P DECODER = 80H - FFH = 128 - 255

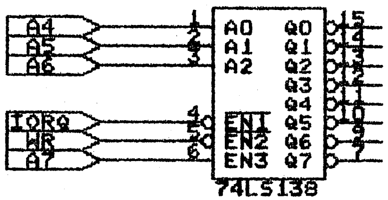
READ ONLY



HEX	DEC
80 - 8F	128 - 143
90 - 9F	144 - 159
A0 - AF	160 - 175
B0 - BF	176 - 191
C0 - CF	192 - 207
D0 - DF	208 - 223
E0 - EF	224 - 239
F0 - FF	240 - 255

16 BIT BLOCK O/P DECODER = 80H - FFH = 128 - 255

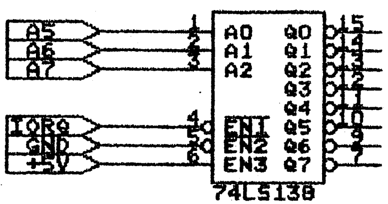
WRITE ONLY



HEX	DEC
80 - 8F	128 - 143
90 - 9F	144 - 159
A0 - AF	160 - 175
B0 - BF	176 - 191
C0 - CF	192 - 207
D0 - DF	208 - 223
E0 - EF	224 - 239
F0 - FF	240 - 255

32 BIT BLOCK I/O DECODER = 00H - FFH = 000 - 255

READ/WRITE



HEX	DEC
00 - 1F	00 - 31
20 - 3F	32 - 63
40 - 5F	64 - 95
60 - 7F	96 - 127
80 - 9F	128 - 159
A0 - BF	160 - 191
C0 - DF	192 - 223
E0 - FF	224 - 255

*** DAVE MITCHELL SOFTWARE ***

E & F WP PATCH 3.3: PATCH 3.3 WRITTEN BY DAVE MITCHELL WILL CONVERT YOUR E & F TAPE WORD PROCESSOR FOR FULL DISK USE WHILE RETAINING ALL ORIGINAL FUNCTIONS. IT ALSO HAS ~~SHIFT LOCK~~ AND PRINTER CONTROL CODES WHICH CAN BE IMBEDDED IN TEXT AND SAVED TO TAPE OR DISK.

BSTWP.F: THIS UTILITY PROVIDED WITH PATCH 3.3 WILL CONVERT BASIC PROGRAMS AND ED/ASS. SOURCE CODE FILES INTO WORD PROCESSOR FILES.

PRICE: AUS/NZ AU$20.00 - UPDATE - AUS-$10.00 - NZ-AU$11.00.

EXTENDED DOS V1.3: THESE COMMANDS ARE AT YOUR DISPOSAL: MERGE, DIRA, DIRA, DIRB, LDIRB, OLD, OLD., DEC, HEX, MENU, CODE, LTAB, MOVE AND UPDATE, STATUSA AND LSTATUSA. STATUSA AND LSTATUSA ALSO WORKS WITH VERSION 1.0 DOS

PRICE: AU$15.00 - POSTAGE INCLUDED

MENU/FILE COPIER: THIS UTILITY WILL READ YOUR DISK DIRECTORY AND PRESENT YOU WITH SEVERAL OPTIONS. USING THE CURSOR YOU CAN RUN/BRUN ANY PROGRAM OR SELECT FILE COPY, REN, ERASE, DRIVE 1 OR 2, ETC. BESIDES COPYING TEXT AND BINARY FILES ALL OTHER FILES CAN BE COPIED AS WELL EXEPT FOR DATA FILES.

PRICE: AU$15.00 - POSTAGE INCLUDED

FOR PURCHASE OR INFORMATION CONTACT:

DAVE MITCHELL 24 ELPHINSTONE STREET
NORTH ROCKHAMPTON QUEENSLAND 4701
AUSTRALIA - PHONE: (079) 27 8519

PETER HICKMAN SOFTWARE - PUBLIC DOMAIN

VZ DISASSEMBLER: WHAT, ANOTHER DISASSEMBLER? BUT, YOU HAVE ALREADY GOT ONE? THIS ONE IS DIFFERENT! THIS PROGRAM IS ENTIRELY WRITTEN IN MACHINE CODE. IT ACTUALLY RUNS ABOUT 40 TIMES FASTER THAN D.S.E.'S DISASSEMBLER (OR ANY ONE ELSE'S). IT WILL DISASSEMBLE ANY PROGRAM THAT YOU CAN BLOAD INTO MEMORY. IT WORKS WITH ANY VZ CONFIGURATION. IT DISASSEMBLES EVEN THE 88 EXTRA Z80 OPCODES THAT ZILOG DOESN'T ADMIT TO.

TAPE AND DISK VERIONS AVAILABLE.

VZ MODEM SOFWARE: DID YOU WANT TO TALK TO OTHER COMPUTERS VIA A MODEM? DID YOU BUY THE DSE TERMINAL EPROM, ONLY TO DISCOVER THAT IT ONLY WORKS WITH TAPE. IT ONLY ALLOWS YOU TO PRINT FILES, NOT SAVE THEM OR SEND THEM!

YOUR PROBLEMS ARE SOLVED! THE HICKMAN BROTHERS, PETER AND ANDREW, HAVE A BRAND NEW PROJECT WHICH WILL ALLOW YOU TO SEND, RECEIVE & SAVE FILES VIA A MODEM. IT WORKS WITH DISK!

INCLUDED ARE INSTRUCTIONS FOR THE HARDWARE MODIFICATIONS. A SMALL MODIFICATION IS NEEDED TO YOUR DISK CONTROLLER. YOUR USER GROUP MAY HELP YOU MODIFY YOUR COMPUTER TO USE THIS EXCITING NEW SOFTWARE!

THE MANUAL IS SUPPLIED ON DISK FOR PRINTING OUT WITH YOUR DISK VERSION OF E & F W/PROCESSOR. IF YOU DO NOT OWN AN E & F W/PROCESSOR THEN PLEASE ENCLOSE ANOTHER $5.00 (TOTAL $30.00) FOR PHOTOCOPYING AND POSTAGE OF THE MANUAL.

PRICE: A NOMINAL FEE TO COVER COSTS. CHECK WITH PETER FOR EXACT AMOUNT.

FOR FURTHER INFORMATION CONTACT:
PETER HICKMAN PO BOX 8 WERRINGTON 2747

** ** CONTRIBUTIONS TO THE JOURNAL ** **

IF YOU ARE THINKING OF CONTRIBUTING TO THE JOURNAL THE PREFERED FORMAT IS BASIC LISTINGS, WORD PROCESSOR OR SOURCE CODE FILES ON TAPE OR DISK. FILES FROM THE FOLLOWING WORD PROCESSORS CAN BE ACCEPTED :-

E & F TAPE OR DISK PATCH 3.1-3.3, WORDPRO CARTRIDGE, WORDPRO PATCH, MOST SOURCE CODE FILES AND ALL QUICKWRITE WORD PROCESSOR FILES.

** ** CLUB MEETINGS - ALL WELCOME ** **

FIRST FRIDAY OF MONTH

** ** FUTURE MEETINGS - NEW VENUE ** **

AS MENTIONED BEFORE WE NO LONGER MEET AT JNC, BUT AT VARIOUS MEMBERS HOMES. MEETINGS WILL BE ONCE A MONTH AS BEFORE WITH THE DATES BEING FIRST FRIDAY OF THE MONTH.

BECAUSE OF SOME LOCAL MEMBERS HAVING TO WORK SHIFTWORK MEETING DATES WILL BE ADJUSTED TO ACCOMODATE THEM. WHETHER YOU ARE A LOCAL MEMBER, INTRA OR INTERSTATE VISITOR PLEASE CHECK WITH JOE LEON FIRST BEFORE COMING OUT.

JOE LEON 33 TIGHES TCE TIGHES HILL 2297 (049) 692 399

* CLUB COMMITTEE & SUBSCRIPTIONS *

PRESIDENT - ROSS WOODS - SECRETARY/EDITOR - JOE LEON
COMMITTEE MEMBERS - COLIN BRIDGE - PETER JONES

SUBSCRIPTION TO - AUST. - 3 ISSUES $11.00 - 6 ISSUES $21.00
H.V.VZ.JOURNAL - N. Z. - 3 ISSUES $13.00 - 6 ISSUES $26.00

FOR MORE INFORMATION CONTACT:

JOE LEON 33 TIGHES TCE TIGHES HILL 2297 (049) 692 399 AUSTRALIA

NOTE : PRICES INCLUDE POST & PACKING

** ** VZ USER GROUPS & PUBLICATIONS ** **

VZ DOWN UNDER - VZ MAGAZINE - 6 ISSUES - $18.00 PER ANUM
HARRY HUGGINS 12 THOMAS SREET MITCHAM VICTORIA 3132

WAVZ - WESTERN AUSTRALIA VZ USER GROUP
GRAEME BYWATER PO BOX 388 MORLEY W A 6062

BRISBANE VZ USERS WORKSHOP - C/O 63 TINGALPA ST. WYNUM WEST 4178
SOFTWARE FOR SALE - DISK MENU

SAPPHIRE PRODUCTIONS - VZ DISK MAGAZINE - PUBLIC DOMAIN
NOTE : VZ DISK MAGAZINE HAS CEASED PRODUCTION

NOTE : WHEN WRITING TO ANY ABOVE OR H.V.VZ. USERS' GROUP FOR INFORMATION PLEASE ENCLOSE A S.S.A.E. OR NZ 2 INT. REPLY COUPONS.